2022

1º Trabalho

Modelação e Programação

REVISÕES, CLASSES E OBJECTOS

DATA DE ENTREGA: 20/03/2022

Os alunos devem colocar as várias resoluções relativas a este trabalho no package tp1. Comece por verificar que já existe, ou então por criar o package tp1.

Avaliação: Este trabalho pode ser realizado individualmente ou em grupo, **com o grupo composto no máximo por dois alunos** (alunos extras serão automaticamente desconsiderados e reprovados). A realização deste trabalho é obrigatória com **nota de aprovação igual ou superior a 10 valores**. Cada alínea deste trabalho tem uma cotação associada listada na própria alínea. Para se obter essa classificação, a aplicação desenvolvida pelo grupo deverá cumprir na íntegra todos os requisitos do enunciado e todos os elementos do grupo deverão conseguir defender o trabalho realizado nessa alínea. Caso essa condição não se verifique, **a cotação que ambos os elementos do grupo terão nessa alínea será zero**. A defesa do trabalho terá que ser feita nas aulas práticas da disciplina, realizadas antes do prazo de entrega.

Relatório: Este trabalho está isento de relatório se o código entregue estiver devidamente comentado seguindo as regras semânticas da ferramenta Javadoc (os outputs desta ferramenta deverão se entregues conjuntamente com o código fonte do trabalho).

Parte A – Revisões

Coloque as alíneas deste grupo no package tp1.pack1Revisoes

A1 (1v) Complete o programa P01CheckPrime que no seu main deverá pedir ao utilizador um número inteiro igual ou superior a 1 (o valor 0 termina a aplicação). De seguida, determina se o valor inserido é um número primo e mostra o resultado na consola. No caso de introdução de valores incorrectos, dever-se-á apresentar uma mensagem de erro e voltar a pedir o número, até o valor estar correcto (por exemplo, se o valor inserido for negativo). Nesta alínea não pode utilizar a instrução "for".

Exemplo:

```
Introduza um número
inteiro:
907
O número 907 é primo
```

A2 (2v) Complete o programa P02FourInaRow de forma a implementar o jogo "4 em linha" em modo consola, com um tabuleiro com capacidade para várias colunas e linhas, de forma a permitir jogar entre dois utilizadores (A e B), e seguindo as indicações dadas no respetivo ficheiro. O programa deverá determinar o seu fim, ou seja, determinar quando um jogador ganhou ou quando

todas as peças estão colocadas e há um empate. O jogo deverá ser suportado por um array bidimensional de caracteres, em que uma posição vazia do tabuleiro corresponderá a um caractere de espaço no array e uma posição com uma peça do jogador A conterá um 'A' (e um 'B' para uma peça do jogador B).

Versão online para quem não conhece o jogo em https://www.coolmath-games.com/0-4-in-a-row.

```
...

Choose a column (Player A): 3
+------------------+
| 0  0  0  0  0  0 |
| 0  0  0  0  0  0 |
| 0  0  0  0  0  0 |
| 0  0  0  0  0  B |
| 0  0  0  0  0  A |
| 0  0  0  B  B  B |
| 0  0  A  A  A  A |
+------------------+

Winner: Player A. Congratulations...
```

Exemplo:

A3 (1v) Na classe P03WorkWithStrings implemente o método **compareStrings**. Este método recebe duas Strings s1 e s2 e procede à sua comparação, devolvendo um valor positivo se s1 for maior que s2, negativo se ao contrário e 0 se iguais. A comparação deve ser feita primeiro em termos lexicográficos caracter a caracter começando pelos caracteres da esquerda para a direita, ou em segundo lugar, segundo o número de caracteres. Se diferentes deve devolver o índice +1/-1 do caractere que faz a diferença. Ex. s1="Bom", s2="Dia", deve devolver -1; s1="Boa", s2="Bom", deve devolver -3; s1="Bom", s2="Bo", deve devolver 3. Uma String a null é considerada menor que uma string não null. Verifique os exemplos disponibilizados no ficheiro **P03WorkWithStrings**.

Parte B – Livros e Coleções de Livros

Coloque as classes deste cenário no package tp1.pack2Livros

B1 (8v) Considere que está a trabalhar para modelar uma pequena aplicação informática para gerir uma biblioteca. Nesta aplicação existe a classe **Livro** que irá conter a descrição de um livro e a classe **Coleccao** que representa uma colecção de livros.
A classe **Livro** descreve um livro com:

- O seu título;
- Número de páginas
- Preço
- Autores

A classe **Coleccao** descreve uma colecção de livros contendo:

- O título da colecção
- Os editores da colecção guardados num arrray sem nulls
- O número de livros existentes na colecção
- Os livros da colecção guardados num array array de dimensão definido na constante MAXOBRAS

Na colecção os livros devem ficar sempre nos menores índices do array onde são guardados e pela ordem de registo.

A Figura 1 apresenta o diagrama de classes UML que descreve estas duas classes. A descrição da funcionalidade dos vários métodos é apresentada nos comentários de documentação dos respetivos métodos nos ficheiros disponibilizados. Os alunos têm de completar o código que está em falta nos diversos métodos de ambas as classes de forma a satisfazer os outputs que são disponibilizados nos ficheiros "Output_Livro.txt" e "Output_Coleccao.txt" e que estão junto ao código que vos foi disponibilizado no moodle. Estes são os outputs dos métodos *main* das respectivas classes **Livro** e **Coleccao**. Encoraja-se a adição de mais testes aos dois métodos *main*.

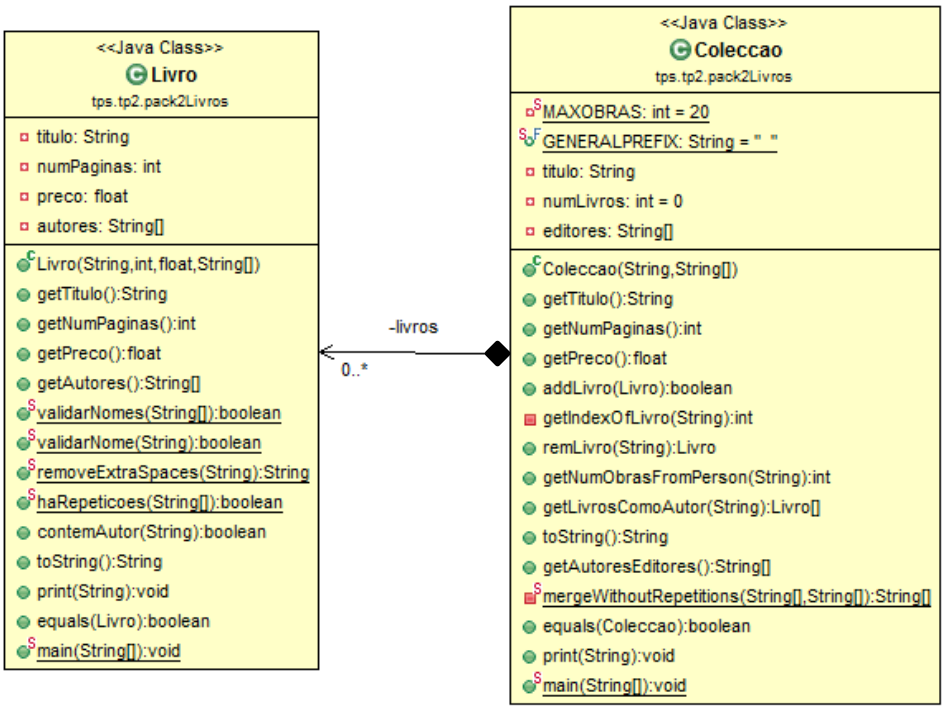

Figura 1 - UML. Diagrama de classes do Livro e Coleccao.

Parte C – Colecções com colecções

Coloque as classes deste cenário no package tp1.pack3Coleccoes

C1 (8v) Faça uma cópia do conteúdo dos métodos da classe **Colecccao** do package pack2Livros para a classe com o mesmo nome deste package. Neste cenário pretende-se que sejam efetuadas as alterações necessárias de modo a que uma colecção possa ter dentro de si também outras colecções e assim sucessivamente, como mostra a Figura 2. Uma colecção deverá suportar as suas colecções num adicional array de **Coleccao** (declarado: *private Coleccao[] coleccoes*) com uma gestão semelhante ao array de livros e por isso conter a variável de **numColeccoes**.

Altere o conteúdo dos métodos existentes na classe **Coleccao** de forma a contemplar a existência de colecções dentro de colecções e de acordo com os comentários de documentação existentes no ficheiro disponibilizado. Complete o método ***main*** de **Coleccao** de modo a permitir testar todas as funcionalidades.

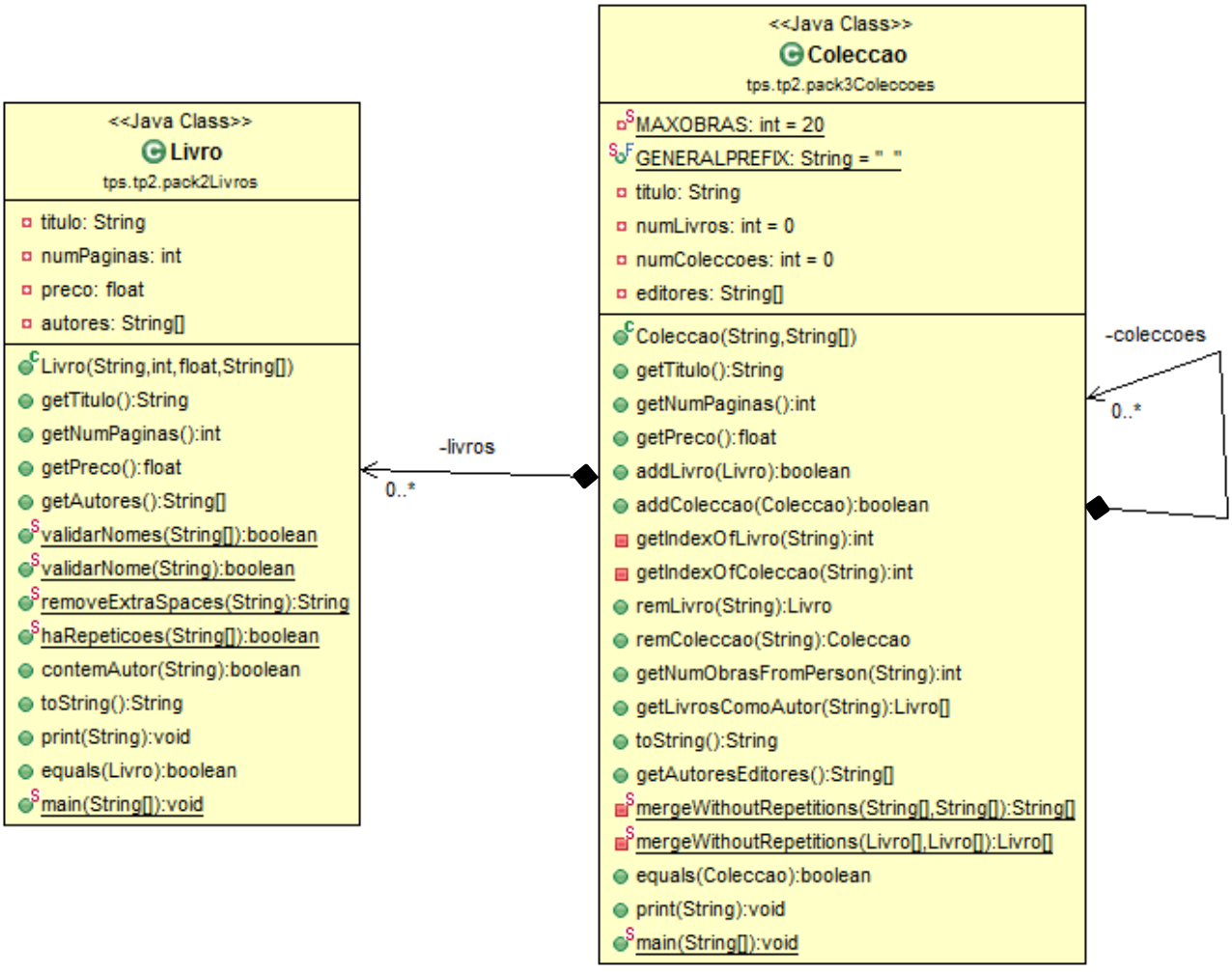

Figura 2 - UML Colecção com colecções.

Em ficheiro, neste package, apresenta-se o output, do método *main*, para a nova classe Coleccao. Encoraja-se a adição de mais testes ao *main*.

O código deste trabalho deve ser entregue até ao dia **20 de março** de 2022.

Bom trabalho, Carlos Júnior, João Ventura, Pedro Fazenda